スーパーコンピュータSX-ACEの利用法

雑誌名	SENAC : 東北大学大型計算機センター広報
巻	48
号	2
ページ	1-28
発行年	2015-04
URL	http://hdl.handle.net/10097/00124871

［大規模科学計算システム］

スーパーコンピュータ SX-ACE の利用法

情報部情報基盤課　共同利用支援係　共同研究支援係
サイバーサイエンスセンター　スーパーコンピューティング研究部

1章　はじめに

本センターはスーパーコンピュータ SX-ACE の運用を 2015 年 2 月から開始しています。本稿では、SX-ACE システムでのプログラミング利用ガイドとして、プログラムの作成からコンパイル、実行等の使い方と利用負担金についてご紹介します。

■　システムの特徴

■ ハードウェア

本センターの SX-ACE システムを図 1 に示します。SX-ACE（日本電気㈱製）は、前システム SX-9 と同じくベクトル並列型スーパーコンピュータを継承しています。今回導入した SX-ACE のシステムは全 2,560 ノードで構成され、1 ノードあたり理論最大演算性能 276GFLOPS、最大ベクトル性能 256GFLOPS の世界初のマルチコア(4 コア)ベクトルプロセッサを 1 基搭載し、システム全体では約 707TFLOPS となります。主記憶は 1 ノードあたり 64GB を搭載し、256GB/s という高いメモリバ ンド幅でプロセッサと接続されることで、高い演算性能とメモリ性能の最適化を実現しています。

図 1 本センターの SX-ACE システム

マルチコアベクトルプロセッサ 1CPU（4 コア）あたり 276 GFLOPS の理論最大演算性能

1 ノードあたり 64GB の共有メモリ

データ転送性能（メモリバンド幅）1CPU あたり 256 GB/s

■ ベクトル計算機

ベクトル計算機とは、ベクトル演算を行う専用のハードウェアを持つコンピュータです。ベクトル演算は、ループ中で繰り返し処理されるような配列データの演算に対して一括して演算を実行するため、高速に演算することができます。

ベクトル計算機は科学技術用の数値計算に適しており、大量のデータを繰り返し処理するような大規模計算に向いていると言われています。本センターでは、流体解析や気象解析、電磁界解析をはじめとする大規模シミュレーションにご利用いただいています。

■ ソフトウェア

UNIX に準拠した SUPER-UX オペレーティングシステムを継承し、ベクトル処理、並列処理に対応した大規模プログラムの実行が可能です。また、SX-ACE に最適化された科学技術計算ライブラリ ASL も引き続きご利用できます。

プログラミング言語

プログラミング言語は Fortran と C/C++を用意しています。それぞれ自動ベクトル化機能、自動並列化機能を有していますので、既存のプログラムを修正することなくベクトル化、並列化することができます。OpenMP や MPI による並列化プログラムも利用可能です。

並列化プログラミング

並列化の方法として表 1 の 3 種類が利用できます。

自動並列化は、単一コアで動作する逐次処理プログラムを、コンパイラが並列化可能な箇所を自動的に判断して並列化します。プログラムを改めて書き直す必要はなく、お持ちのプログラムをそのまま利用することができます。

OpenMP は、自動並列化と同じく逐次処理プログラムを並列化します。並列化の判断は自動ではなくユーザが行います。ソース中の並列化したい箇所に並列化指示行を追加します。

MPI は、メッセージ交換用ライブラリによりプロセッサ間通信を行います。データの分割、処理方法等の並列処理手順を明示的に記述した MPI プログラムを作成する必要があります。

ノード内では自動並列化あるいは OpenMP による並列処理を、複数のノードを利用する場合は MPI による並列処理を用います。したがって、自動並列化や OpenMP は 4 並列までのノード内並列処理ができます。それ以上の並列数で実行する場合は、複数のノードを必要とするので MPI による並列処理を行います。また、大規模なメモリを必要とするプログラムも、MPI を利用して複数のノードで実行する必要があります。

本センターでは最大 1,024 ノード[1]を利用した 4,096 並列での実行が可能です。

自動並列化と OpenMP によるノード内並列処理手順は 3 章で、MPI による並列処理手順は 4 章で、プログラムの実行方法は 5 章で紹介します。

表 1 並列化の種類、特徴、並列数、メモリ量

並列化の種類	逐次処理プログラムの並列化	並列化の指示	最大並列数	最大メモリ量
自動並列化	そのままで可能	自動 (コンパイラ)	4 並列 ノード内	60GB
OpenMP	指示行を追記することで可能	手動 (ユーザ)	4 並列 ノード内	60GB
MPI	MPI 用プログラムに変更する	MPI ライブラリを用いて プログラミング (ユーザ)	4,096 並列 1,024 ノード	60GB ×ノード数

[1] 513 ノード以上の利用は、2015 年後半を予定しています。

2 章　システムの構成

本センターでは、大型計算機として SX-ACE システム 2,560 ノードと並列コンピュータシステム（LX 406Re-2）68 ノードの 2 つのシステムをサービスしています（図 2）。

並列コンピュータはスーパーコンピュータのフロントエンドサーバの役割も担っています。

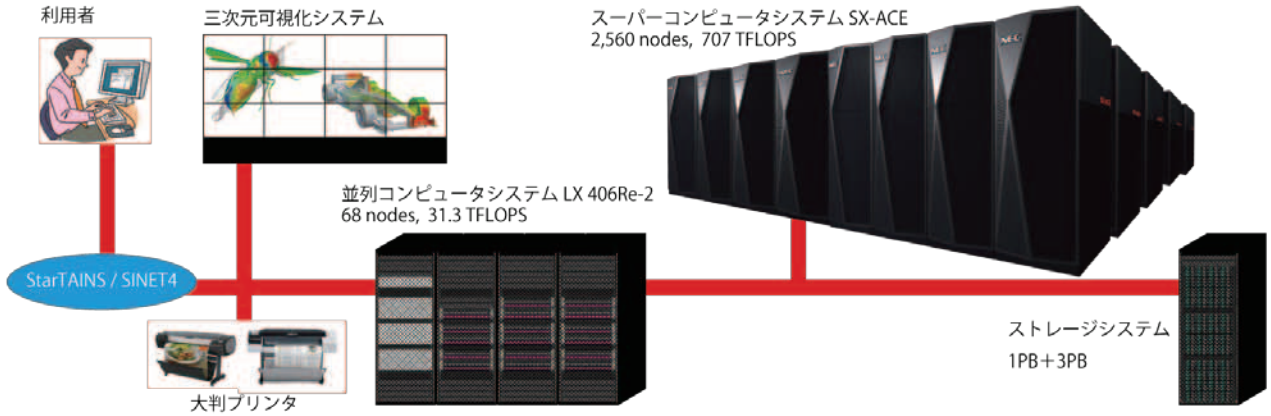

図 2 システム構成図

3 章　プログラミング　～逐次処理、共有メモリ並列処理 ～

本章では、単一のコアで実行する逐次処理と、自動並列化および OpenMP による共有メモリ並列処理について利用手順を紹介します。

MPI による並列化プログラミング手順については、4 章で紹介しますが、コンパイルコマンドに違いがある以外は、同じ手順ですのでこの章と合わせてご覧ください。

■　フロントエンドサーバ（並列コンピュータ）へのログイン

SX-ACE 用のプログラミングは、並列コンピュータ上で全ての作業を行えるようにしています。ソースファイル作成、コンパイル、実行など一連の作業が可能です。SX-ACE システムのパフォーマンスを演算に専念させるために、フロントエンドサーバとの 2 段構成としています（図 3）。

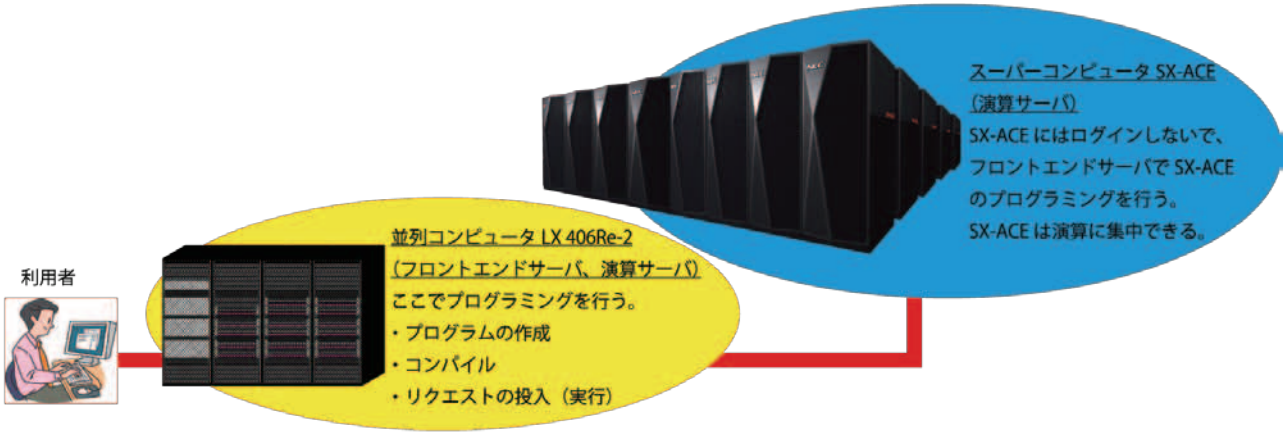

図 3 フロントエンドサーバと演算サーバ

■ ログイン

作業を行うため並列コンピュータにログインします。リモート接続は、ssh コマンドまたは SSH 対応リモート接続ソフト[2]をご利用ください。

並列コンピュータの OS は Linux です。公開鍵暗号方式による認証のみ利用できます[3]。アカウント希望の場合は、共同利用支援係に利用申請し利用者番号と初期パスワードを発行してもらいます。

並列コンピュータへの初回ログイン時には公開鍵と秘密鍵のペアを作成する必要があります。鍵ペアの作成方法については本誌 105 ページの「SSH アクセス認証鍵生成サーバの利用方法」をご参照ください。

なお、他人名義の利用者番号でのシステム利用は禁止します。パスワード、秘密鍵、パスフレーズの使い回しは、不正アクセスのリスク（不正ログイン、クライアントのなりすまし、暗号化された通信の暴露、他サーバへの攻撃等）が非常に高く、大変危険です。利用者登録を行うことによる年間維持費等は発生しませんので、利用される方はそれぞれで利用申請をお願いいたします。

並列コンピュータホスト名

```
front.cc.tohoku.ac.jp
```

リスト 1 ssh コマンドによる接続例

```
localhost$ ssh -i ~/.ssh/id_rsa 利用者番号@front.cc.tohoku.ac.jp
Enter passphrase for key '/home/localname/.ssh/id_rsa':パスフレーズを入力
（初回接続時のメッセージ）： yes を入力
front1$（コマンド待ち状態）
```

暗号鍵は複数の端末や他人と共有してはいけません。また、メールに暗号鍵を添付したり USB メモリにコピーしたりすると不正アクセスのリスクとなります。並列コンピュータにログインする端末を追加する場合は、その端末で新規に鍵ペアを作成し、authorized_keys に追加登録してください。

ログイン端末の追加方法

ログインする端末を追加する場合は、追加する端末で鍵ペアの作成を行います。必ずパスフレーズの設定を行って鍵を作成して下さい。作成した公開鍵を既に並列コンピュータにログイン出来る端末に転送します。公開鍵ですので転送方法はメール本文に記載しても構いません。

作成した公開鍵の内容を利用者のホームディレクトリのファイル（~/.ssh/authorized_keys）に追記します。接続元が Linux、OS X の場合の鍵の作成方法をリスト 2 に示します。

リスト 2 公開鍵と秘密鍵の作成方法

```
【利用する端末で鍵ペアを作成】
localhost $ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/localname/.ssh/id_rsa):（ファイル名を指定）

Enter passphrase (empty for no passphrase):（必ずパスフレーズを設定）
Enter same passphrase again:（同じパスフレーズを入力）

指定した場所(/home/localname/.ssh)に鍵ペア(暗号鍵:id_rsa 公開鍵:id_rsa.pub)が生成される

【作成した公開鍵を既に並列コンピュータにログイン出来る端末に転送】

【作成した公開鍵を並列コンピュータに転送】
localhost $ scp /home/localname/.ssh/id_rsa.pub 利用者番号@front.cc.tohoku.ac.jp:~
```

[2] Windows であれば、TeraTerm 等のフリーソフトが利用できます。

[3] パスワード認証方式は 2015 年 4 月 13 日で廃止しました。

```
（パスフレーズを入力）

【公開鍵を追記登録】
front1 $ cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
front1 $ exit
```

~/.ssh/authorized_keys を削除すると、全ての暗号鍵からのログインが出来なくなります。センターではセキュリティインシデントに対する緊急対応として、全ユーザの~/.ssh/authorized_keys を削除する場合があります。

■ パスワードの変更

パスワードの変更は passwd コマンドで行います。パスワードの変更方法をリスト 3 に示します。
コマンドを実行すると、並列コンピュータ、可視化サーバ、プリンタサーバ、およびファイル転送サーバのログインパスワードが変更されます。入力したパスワードは表示されません。

リスト 3 パスワードの変更方法

```
front1 $ passwd
ユーザー 利用者番号 のパスワードを変更。
Enter login(LDAP) password: （現在のパスワードを入力）
新しいパスワード: （新しいパスワードを入力）
新しいパスワードを再入力してください: （新しいパスワードを入力）
LDAP password information changed for 利用者番号
passwd: 全ての認証トークンが正しく更新できました。
```

■ ログインシェルの確認と変更

ログインシェルの確認と変更は fchsh コマンドで行います。ログインシェルの確認方法と変更方法をリスト 4 に示します。
ログインシェルの変更が SX-ACE と LX 406Re-2 に反映されるまで 15 分程度かかります。また、SX-ACE で利用可能なシェル以外をログインシェルに設定した場合、リクエストが正常に実行されなくなります。

リスト 4 ログインシェルの確認方法と変更方法

```
front1 $ fchsh（ログインシェルの確認）
Enter Password:（パスワードを入力）
loginShell: /bin/tcsh（現在のログインシェルが表示される）

front1 $ fchsh /bin/bash（ログインシェルを/bin/bash に変更）
Enter Password:（パスワードを入力）
Changed loginShell to /bin/bash（ログインシェルが変更された）
```

■ ホームディレクトリ

ホームディレクトリは、プログラムファイル等を置く自分専用のディスク領域です。ディレクトリ名は、/uhome/利用者番号です。利用者番号作成時の容量制限は 1TB です。ファイル容量の追加申請によりディスク領域を増やすことも可能です。ホームディレクトリはスーパーコンピュータシステムと並列コンピュータシステムで共有しています。

ホームディレクトリ

```
/uhome/利用者番号
```

■　プログラムの作成とコンパイル

■ ソースファイルの作成

並列コンピュータ上でソースファイルを作成するためのエディタは、emacs または vi が一般的です。

研究室等のサーバ、パソコンから転送する場合は、SSH 対応のファイル転送ソフト[4]で並列コンピュータのホームディレクトリに転送してください。その際、ソース(テキスト)ファイルは ASCII モードで転送します。

Windows 環境で作成したソースコードの改行コードを Linux 用にするためには dos2unix コマンドを利用します。また、並列コンピュータの文字コードは UTF-8 に設定されていますので、ソースコード内で日本語を利用する際は文字コードを合わせてください。並列コンピュータでは文字コードの変換に nkf コマンドが利用できます(リスト 5)。

リスト 5 dos2unix コマンドと nkf コマンドの利用例

```
front1 $ dos2unix source.f90                  (改行コードを Linux に合わせる)
front1 $ nkf -w --overwrite source.f90        (文字コードを UTF-8 に変換)
```

■ コンパイラ

SX-ACE 用に最適化されたコンパイラ Fortran および C/C++を用意しています。両言語とも並列コンピュータ上でコンパイルとリンクを行います。

● Fortran コンパイラの仕様

FORTRAN90/SX（日本電気製）
ISO/IEC 1539-1:1997 準拠
自動ベクトル化、自動並列化、OpenMP 対応

● C/C++コンパイラの仕様

C++/SX（日本電気製）
ISO/IEC 9899:1999 C 準拠
ISO/IEC 14882:2003 C++ 準拠
自動ベクトル化、自動並列化、OpenMP 対応

■ コンパイルを行う

通常のコンパイルと同様に、それぞれの言語用コマンドでコンパイルします。この項目では単一コアで実行する逐次処理と、自動並列化または OpenMP による並列処理のコンパイル手順について解説します。

Fortran プログラムのコンパイル

sxf90 コマンドでコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

ソースファイルの拡張子は、自由形式(フリーフォーマット)なら.f90 か.F90、固定形式(7カラム目から記述)なら.f か.F を付けます(表 3)。

並列化コンパイルは、ここで並列数を意識する必要はありません。実行する時点で希望する並列数を環境変数で指定します。ノード内並列数は自動並列の場合は環境変数 F_RSVTASK で指定し、OpenMP 並列の場合は環境変数 OMP_NUM_THREADS で指定します。詳細は 5 章で説明します。

[4] Windows であれば、WinSCP 等のフリーソフトがあります。

● **コンパイル【逐次処理】**

```
front1 $ sxf90 オプション ソースファイル名
```

● **コンパイル【自動並列化】**

```
front1 $ sxf90 -Pauto オプション ソースファイル名
```

・OpenMP プログラムなら、-Pauto の箇所を-Popenmp にします。

表 2 は、主なオプションです。詳細は、sxman sxf90 コマンドでご覧ください。

表 2 主なオプション

オプション	機能
-Pauto	自動並列化機能を利用する。
-Popenmp	OpenMP 対応機能を利用する。
-R5	ベクトル化／並列化状況を表示した編集リストを出力する。
-ftrace	簡易性能解析機能を利用する。
-eC	配列要素参照時、添字の値が範囲内であるか検査する（バウンズチェック）。

表 3 拡張子とフォーマット

拡張子	フォーマット
.f90 .F90	自由形式
.f .F	固定形式

C/C++プログラムのコンパイル

sxcc コマンドで C プログラムを、sxc++コマンドで C++プログラムをコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

並列化コンパイルは、ここで並列数を意識する必要はありません。実行する時点で希望する並列数を環境変数で指定します。ノード内並列数は自動並列の場合は環境変数 F_RSVTASK で指定し、OpenMP 並列の場合は環境変数 OMP_NUM_THREADS で指定します。詳細は 5 章で説明します。

● **コンパイル【逐次処理】**

```
front1 $ sxcc  オプション ソースファイル名
front1 $ sxc++ オプション ソースファイル名
```

● **コンパイル【自動並列化】**

```
front1 $ sxcc -Pauto  オプション ソースファイル名
front1 $ sxc++ -Pauto オプション ソースファイル名
```

・OpenMP プログラムなら、-Pauto の箇所を-Popenmp にします。

表 4 は、主なオプションです。詳細は、sxman sxcc コマンド、または sxman sxc++コマンドでご確認ください。

表 4 主なオプション

オプション	機能
-Pauto	自動並列化機能を利用する。
-Popenmp	OpenMP 対応機能を利用する。
-size_t64	4G バイト以上の配列、構造体を利用する。
-Rfmtlist	ベクトル化／並列化状況を表示した編集リストを出力する。
-ftrace	簡易性能解析機能を利用する。

表 5 拡張子とフォーマット

拡張子	言語
.c	C
.cpp	C++

逐次処理、自動並列化または OpenMP により並列化されたプログラムは、1 ノードのみ利用できます。複数ノードを利用する場合は MPI による並列化が必要です。MPI については次章をご参照ください。

4 章 プログラミング ～ MPI 並列処理 ～

本章では、MPI による並列化プログラミング手順について紹介します。基本的な手順は前章と同じですので、コンパイルコマンドの異なる点について紹介します。

■ コンパイルを行う

MPI プログラムは、MPI 用コマンド sxmpif90、sxmpic++、sxmpicc コマンドでコンパイルします。

MPI 並列 Fortran プログラムのコンパイル

MPI 並列の Fortran コードは sxmpif90 コマンドでコンパイルします。利用したい機能があればオプションと、ソースファイル名を指定します。

```
front1 $ sxmpif90 オプション ソースファイル名
```

・オプションは、sxf90 コマンドと共通です。sxman sxf90 コマンドでご覧ください。

MPI 並列 C/C++プログラムのコンパイル

MPI 並列の C コードは sxmpicc コマンドで、MPI 並列の C++コードは sxmpic++コマンドでコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

```
front1 $ sxmpicc  オプション ソースファイル名
front1 $ sxmpic++ オプション ソースファイル名
```

・オプションは、sxcc コマンド、sxc++コマンドと共通です。詳細は sxman sxcc コマンド、sxman sxc++コマンドでご確認ください。

■ ハイブリッド並列

MPI と自動並列、または MPI と OpenMP を組み合わせたハイブリッド並列と呼ばれる並列化手法があります(図 4)。ノード間は MPI 並列化し、ノード内は自動並列化や OpenMP による並列化する形をとります。

例えば、512 ノードを使ったハイブリッド並列を考えた場合、512 プロセスで実行する MPI プログラムを作成します。さらに、自動並列化オプション-Pauto をつけコンパイルすると、MPI と自動並列のハイブリッド並列用オブジェクトが作成されます。ノード間は MPI の 512 並列、ノード内は自動並列化の 4 並列の、計 2,048 並列で実行できます。実行方法についての詳細は次章で説明します。

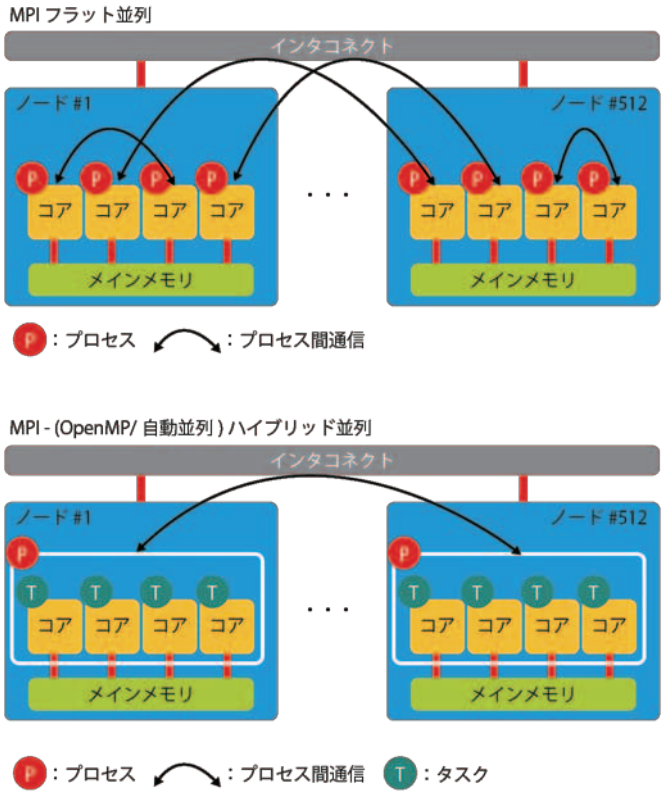

図 4 MPI フラット並列とハイブリッド並列

```
front1 $ sxmpif90 -Pauto オプション ソースファイル名
```

• ノード内が OpenMP 並列なら、-Popenmp オプションを使います。

5 章　バッチリクエスト

■　バッチリクエストの概要

フロントエンドサーバでコンパイル作業を行って作成したプログラムの実行は、バッチリクエストと呼ばれる方法で計算機に実行を依頼します。本センターではバッチリクエストの処理に NQS Ⅱを採用しています。以下では NQS Ⅱによるバッチリクエストについて説明します。

始めに NQS Ⅱにプログラムの実行を依頼するため、実行の手続きを書いたバッチリクエストファイルを作成します。このファイルを NQS Ⅱに投入することで、プログラムの実行が可能になります（図 5）。利用者は実行する計算機とノード数を指定して投入します。

順番が回ってくると NQS Ⅱは自動的にリクエストを実行します。リクエストはバックフィルスケジュール機能で実行順序が制御されています。利用者が実行時間を実行時間制限の規定値以下に指定すると、利用ノード数が少ない場合ノードの空き状況によっては実行開始時間が早くなることがあります。

リクエスト投入後は、リクエストの状態の確認、また投入済みのリクエストをキャンセルすることも可能です。プログラムの実行が終了するとリクエストは NQS Ⅱの情報から消え、標準出力ファイルと標準エラー出力ファイルが出力されます。

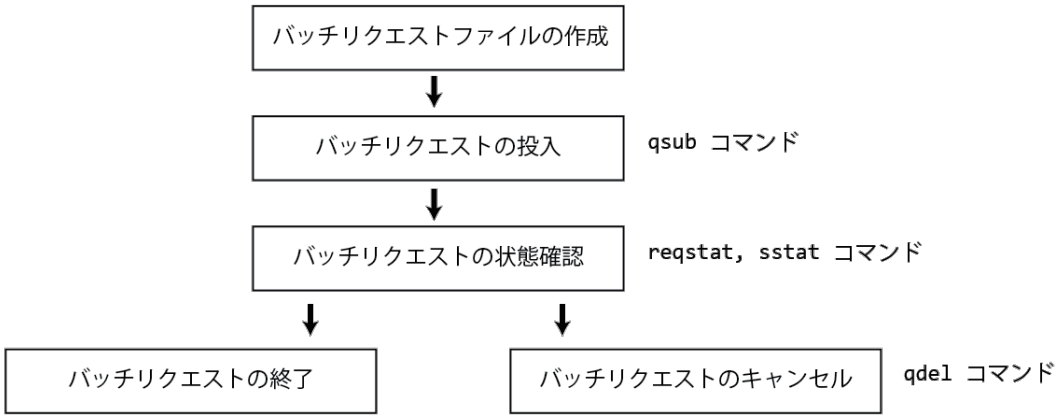

図 5 バッチリクエストの流れ

■　バッチリクエストファイルの作成

プログラムの実行手続きを、通常のシェルスクリプトと同じ形式で記述します。csh スクリプトと sh スクリプト、どちらでも記述できます(以降、解説は csh スクリプトで記述します)。適当なファイル名を付け作成します。以降ではバッチリクエストファイル名を run.csh とします。

基本的に必要となるのは、実行するマシンとノード数の指定、ホームディレクトリから作業ディレクトリへ移動、プログラムの実行、です。他に環境変数の指定、ファイルの操作コマンド等があれば適切な箇所に手続きを記述します。

SX-ACE で実行する場合の利用形態と必須オプションを表 6 に示します。通常の利用の場合、-q の後に sx を指定し、-b の後に利用ノード数を指定します。

実行時間の設定は、-l elapstim_req=hh:mm:ss で設定します。通常利用の場合、実行時間制限の規定値でリクエストは自動的に終了します。実行時間が規定値を超えるリクエストは、必ず実行時間を指定してください。実行時間は最大値まで設定が可能です。その他のオプションは表 7 をご参照ください。

表 6 SX-ACE の利用形態と-q および-b オプション

利用形態	利用ノード数[5]	実行時間制限 (経過時間)	メモリサイズ制限	-q オプション	-b オプション
通常	1～256	規定値： 2 週間 最大値： 1 ヶ月	60GB×ノード数	sx	利用ノード数
	257～1,024	規定値： 1 ヶ月 最大値： 1 ヶ月			
無料	1	1 時間	60GB	sx	f
デバッグ用	1～16	2 時間	60GB×ノード数	debug	利用ノード数
	17～32	24 時間			

表 7 qsub コマンドの主なオプション

オプション	機能
-q(必須)	計算機名 sx または debug を指定します。
-b(必須)	利用ノード数または f を指定します。

[5] 513 ノード以上の利用は、2015 年後半を予定しています。

-A	課金先のプロジェクトコードを指定します。指定が無ければデフォルトのプロジェクトコードに課金されます。
-N	リクエスト名を指定します。指定がなければ、リクエストファイル名がリクエスト名になります。
-o	標準出力のファイル名を指定します。指定がなければ、リクエスト投入時のディレクトリに「リクエスト名.o リクエスト ID の番号」のファイル名で出力されます。
-e	標準エラー出力のファイル名を指定します。指定がなければ、リクエスト投入時のディレクトリに「リクエスト名.e リクエスト ID の番号」のファイル名で出力されます。
-jo	標準エラー出力を標準出力と同じファイルへ出力します。
-l elapstim_req=hh:mm:ss	最大経過時間を指定します。設定時間は、時:分:秒を hh:mm:ss の形式で指定します。
-m b	リクエストの処理が開始したときにメールが送られます。
-m e	リクエストの処理が終了したときにメールが送られます。
-M メールアドレス	メールの送信先を指定します。指定がなければ、「利用者番号@front.cc.tohoku.ac.jp」宛に送られます。

● 逐次プログラム、自動並列/OpenMP 並列の場合

リスト 6 は逐次プログラムを実行する場合のバッチリクエストファイルの一例です。ホームディレクトリ直下 work ディレクトリの a.out を実行する手続きを記述しています。

リスト 6 バッチリクエストファイル例

```
# test job.1              コメント行
#PBS -q sx -b 1          #実行マシンとノード数を指定
cd work                  #実行ディレクトリへ移動
./a.out                  #実行形式ファイルを指定
```

- 1行目:# 以降はコメントです。動作には影響しません。
- 2行目:SX-ACE の 1 ノードを使う指定です。
- 3行目:cd work で作業ディレクトリ(実行形式ファイルのあるディレクトリ)へ移動します。省略するとホームディレクトリを指定したことになります。
- 4行目:a.out はコンパイルして作成した実行形式ファイル名です。あらかじめコンパイルし作成しておきます。自動並列や OpenMP による並列処理も、同じ形式で指定します。

参考 1 : 作業ディレクトリの指定

NQSⅡ用の環境変数のひとつに PBS_O_WORKDIR 変数があります。この変数には、qsub コマンドを実行した時点のカレントディレクトリが設定されます。

つまり、work ディレクトリでこのリクエストを投入する(qsub を実行する)と、$PBS_O_WORKDIR にはカレントディレクトリの work が設定され cd work と同じことになります。PBS_O_WORKDIR 変数を設定することで、ディレクトリの具体名を記述する必要がなくなります。

リスト 7 環境変数 PBS_O_WORKDIR

```
# test job.2              コメント行
#PBS -q sx -b 1          #実行マシンとノード数を指定
cd $PBS_O_WORKDIR        #実行ディレクトリを環境変数で指定
./a.out                  #実行形式ファイルを指定
```

参考 2：qsub コマンドオプションの埋め込み

表 7 に qsub コマンドの主なオプションを示します。-q と-b オプションは必須指定です。

qsub コマンドのオプションはバッチリクエストファイルに埋め込みます（リスト 8）。設定方法は、最初のコマンドより前の行に、#PBS という文字列を先頭に指定します。#PBS の後に空白を一文字以上入れ、指定したいオプションを続けます。一行に複数のオプション指定も可能です。

リスト 8 の例は、2 行目で実行計算機に SX-ACE を指定(-q sx)と、利用ノード数 1 を指定（-b 1）し、3 行目で標準エラー出力を標準出力ファイルにひとまとめにし(-jo)、4 行目でリクエスト名を test04 に指定(-N test04)し、5 行目で最大経過時間を 1 時間（-l elapstim_req=01:00:00）に、それぞれ指定しています。

埋め込みオプションとコマンド列に同じオプションを指定した場合は、コマンド列の方が優先されます。

リスト 8 オプションの埋め込み

```
# test job.4
#PBS -q sx -b 1
#PBS -jo                          #標準エラー出力を標準出力と同じファイルへ出力
#PBS -N test04                    #リクエスト名を test04 にする
#PBS -l elapstim_req=01:00:00     #最大経過時間を 1 時間に設定
cd $PBS_O_WORKDIR
./a.out
```

● MPI 並列の場合

MPI プログラムの場合は mpirun コマンドにオプションと実行形式ファイル名を指定します。mpirun コマンドのオプションに -np と -nn の指定が必要です。-np オプションには総 MPI プロセス数、-nn には利用するノード数を指定します。-b で指定する値と、-nn で指定する値は必ず同数に指定します。SX-ACE は 1 ノードに 4 コアを搭載していますので、-np には -b の 4 倍の数まで指定できます。

なお、バッチリクエストファイル中で指定した環境変数は、MPIEXPORT のあとにダブルクオーテーション中に記述し、全てのノードに設定を反映させる必要があります。

MPI 並列プログラムを 32 ノードを利用して実行する例をリスト 9 に示します。

```
mpirun オプション ./実行形式ファイル名
```

表 8 mpirun コマンドのオプション（必須）

オプション	引数
-np	総 MPI プロセス数
-nn	使用ノード数（-b と同数を指定）

リスト 9 MPI 並列実行（32 ノード利用）

```
# test job.3
#PBS -q sx -b 32                  #SX-ACE の 32 ノードを利用する
cd $PBS_O_WORKDIR
mpirun -np 128 -nn 32 ./a.out     #128MPI プロセスで実行する
```

● ハイブリッド並列の場合

MPI と自動並列/OpenMP 並列されたハイブリッド並列のプログラムを実行する場合、ノード内の並列数を環境変数で指定する必要があります。SX-ACE では自動並列数と OpenMP 並列数を指定する環境変数が異なりますのでご注意ください（表 9）。

MPI と自動並列のプログラムをハイブリッド並列で実行する例を、リスト 10 に示します。

表 9 ノード内並列数を指定する環境変数

並列方法		環境変数
自動並列(-Pauto)	Fortran プログラムの場合	F_RSVTASK
	C/C++プログラムの場合	C_RSVTASK
OpenMP 並列(-Popenmp)		OMP_NUM_THREADS

リスト 10 ハイブリッド並列実行(32 ノード利用、自動並列)

```
# test job.4
#PBS –q sx -b 32                #SX-ACE の 32 ノードを利用する
cd $PBS_O_WORKDIR
setenv F_RSVTASK 4              #ノード内は自動並列で 4 並列実行
setenv MPIEXPORT “F_RSVTASK”    #全てのノードに環境変数を設定
mpirun –np 32 –nn 32 ./a.out    #32MPI プロセス x4 並列＝128 並列で Fortran プログラムを実行する
```

● 標準入出力ファイルの指定

1 つのノードでの実行では、標準入出力ファイルをリダイレクション記号(<, >)で指定することができますが、MPI プログラムの実行ではこれらのリダイレクション記号は利用できません。

この場合、標準入力ファイルの指定は環境変数 F_FF05 で行います(リスト 11)。標準出力および標準エラー出力は、MPI 並列数分複数出力されます。このとき mpisep.sh を利用すると、同一ファイルにプロセス毎の出力が入り混じって出力されないように、MPI プロセスごとにファイルを分けて出力することができます(リスト 12)。シェルスクリプト /usr/lib/mpi/mpisep.sh を実行形式ファイル a.out の前に記述し、環境変数 MPISEPSELECT に 1～4 の値を指定することで、表 10 の動作を選択します。

リスト 11 標準入出力ファイルの指定例

```
# test job.5
#PBS –q sx -b 32
setenv F_FF05 infile            #標準入力は装置番号 05
setenv MPIEXPORT “F_FF05”       #全てのノードに環境変数を設定
cd $PBS_O_WORKDIR
mpirun –np 32 –nn 32 ./a.out
```

リスト 12 標準出力、標準エラーファイルをプロセス毎に分割

```
# test job.6
#PBS -q sx -b 32
setenv MPISEPSELECT 4           #出力形式を 4 に設定
setenv MPIEXPORT “MPISEPSELECT” #全てのノードに環境変数を設定
cd $PBS_O_WORKDIR
mpirun -np 32 -nn 32 /usr/lib/mpi/mpisep.sh ./a.out
```

表 10 MPISEPSELECT の引数と動作

MPISEPSELECT の引数	動作
1	各 MPI プロセスの標準出力のみを stdout.uuu:rrr へ格納
2	各 MPI プロセスの標準エラーのみを stderr.uuu:rrr へ格納(規定値)
3	各 MPI プロセスの標準出力と標準エラーを各々stdout.uuu:rrr および stderr.uuu:rrr へ格納
4	各 MPI プロセスの標準出力と標準エラーを同一ファイル stdout.uuu:rrr へ格納

・実行時、stdout.*や stderr.*の同名ファイルが存在する場合は、上書きでなく追加出力します。

● MPI 用実行性能情報

MPIPROGINF 環境変数の設定により、MPI プロセス毎のプログラム性能情報(PROGINFO)の表示形式を変更することができます。DETAIL は Min/Max/Ave にまとめたもの、ALL_DETAIL は全てのプロセスについて、それぞれ性能情報を表示します。性能情報はプログラム実行終了後、標準エラー出力ファイルに出力されます。

表 11 MPIPROGINF の引数と動作

MPIPROGINF の引数	動作
DETAIL	性能情報を集約形式で出力します。
ALL_DETAIL	性能情報を拡張形式で出力します。 全ランクの情報を出力します。

■　バッチリクエストの投入

プログラムの実行は、作成したバッチリクエストファイルを NQSⅡに投入することで行います。

```
front1 $ qsub オプション　バッチリクエストファイル名
```

リクエストが正常に投入されると、システムからのメッセージが返ります(リスト 13)。1234.job1 がリクエスト ID で、リクエストの状況確認やキャンセル等、リクエストの操作の際に指定が必要になります。

リスト 13 qsub コマンド実行時のメッセージ例

```
front1$ qsub run.csh
Request 1234.job1 submitted to queue : sx32.
```

■ バッチリクエストの状態確認

● reqstat コマンド

reqstat コマンドは、投入されたリクエストの状態を表示します(
リスト 14)。状態は STATE 項目に表示されます(表 13)。リクエストはバッチサーバ(job1 または job2)毎に出力されます。

リソースに空きがなければ実行待ち状態になり、順番が回ってくると自動的に実行状態に入ります。システム内に自分のリクエストが存在しない場合は、「No request.」と表示されます(リスト 15)。

リスト 14 reqstat コマンド例

```
front1$ reqstat
statistics sampled at 2015/02/20 19:18:01 in job1.
REQUEST ID      USER     GROUP    QUEUE    NODE ELAPS     STATE       TIMES             REQUEST NAME
--------------- -------- -------- -------- ---- --------- ----------- ----------------- ---------------
1733.job1       利用者番号 users    sx64      64   1:00:00  running     15/02/20 12:00:00 test01
1735.job1       利用者番号 users    sx64      64   1:00:00  queued      15/02/20 18:55:00 test03

statistics sampled at 2015/02/20 19:18:01 in job2.
REQUEST ID      USER     GROUP    QUEUE    NODE ELAPS     STATE       TIMES             REQUEST NAME
--------------- -------- -------- -------- ---- --------- ----------- ----------------- ---------------
1734.job2       利用者番号 users    sx64      64   0:00:00  wait        15/02/20 13:55:00 test02
```

表 12 reqstat コマンドの主な表示項目

項目名	内容
RequestID	リクエスト ID
USER	利用者番号
GROUP	利用者の所属グループ
QUEUE	キュー名
NODE	利用ノード数
ELAPS	経過時間制限
STATE	リクエストのステータス
TIMES	ステータスが変化した日時
REQUEST NAME	-N で指定したリクエスト名もしくはバッチリクエストファイル名

表 13 STATE 項目の主な表示とリクエスト状態

表示	リクエスト状態
wait	実行ノードの決定待ち
queued	実行ノードが決まり、実行順待ち
running	実行中

リスト 15 投入したリクエストがない場合

```
front1 $ reqstat
No request.
```

● **sstat コマンド**

sstat コマンドは投入されたリクエストの実行開始予定時刻を表示します(リスト 16)。ただし、255 ノード以下の利用の場合のみ表示されます。利用ノード数が少ない場合バックフィルスケジュール機能により、予定実行開始時刻が早まることがあります。-l elapstim_req オプションで指定した時間が短いほど、待ちリクエストの隙間にリクエストの実行が割り当てられる可能性が大きくなりますので、必要十分な実行時間を指定することをお勧めいたします。

リスト 16 sstat コマンド例

```
front1 $ sstat

RequestID       ReqName  UserName Queue                  Pri STT PlannedStartTime
--------------- -------- -------- -------- ----------------- --- -------------------
2512.job1       test03   利用者番号 sx32       0.5002/  0.5002 ASG 2015-02-20 08:28:20
2513.job1       test04   利用者番号 sx32       0.5002/  0.5002 ASG 2015-02-20 09:28:20
```

表 14 sstat コマンドの表示項目

項目名	内容
RequestID	リクエスト ID
ReqName	-N で指定したリクエスト名 指定がなければ、リクエストファイル名がリクエスト名になります。
UserName	利用者番号
QUEUE	キュー名
Pri	優先度
STT	リクエストのステータス
PlannedStartTime	予定実行開始時刻

■ バッチリクエストのキャンセル

投入したリクエストの削除、または実行中のリクエストを停止する場合は、qdel コマンドにリクエスト ID を指定します(リスト 17)。リクエスト投入時、または reqstat コマンドで表示されるリクエスト ID をバッチサーバ名まで指定してください。

リスト 17 qdel コマンド例

```
front1 $ qdel 1234.job1
Request 1234.job1 was deleted.
```

6 章　スーパーコンピュータ用ライブラリ

■　科学技術計算ライブラリ ASL

ASL(Advanced Scientific Library)は、科学技術計算の広範な分野の数値シミュレーションプログラムの作成を強力に支援する FORTRAN 版数学ライブラリです。ASL を用いることによって、難解な数値計算アルゴリズムの詳細に煩わされること無く高度な科学技術計算プログラムを作成することができます。また、ASL C 言語インタフェース(Advanced Scientific Library C INTerface)は、C/C++プログラムから FORTRAN 版 ASL を容易に利用できるように作成されたインタフェースライブラリです。

ASL は次のような数値計算分野に対応しています。

＜基本機能＞
　格納モードの変換、基本行列演算、最小二乗法、固有値・固有ベクトル、連立 1 次方程式(直接法)、連立 1 次方程式(反復法)、対称連立一次方程式(反復法)、非対称連立一次方程式(反復法)、フーリエ変換とその応用／時系列分析、微分方程式とその応用、数値微分、数値積分、3 次元境界要素法用の数値積分法、近似・補間、スプライン関数、特殊関数、乱数、ソート・順位付け、方程式の根、極値問題・最適化
＜統計機能＞
　確率分布、基礎統計量、推定と検定、分散分析・実験計画、ノンパラメトリック検定、多変量解析、フーリエ変換とその応用／時系列分析、近似・回帰分析、乱数、ソート・順位付け
＜共有メモリ並列処理機能＞
　基本行列演算、連立 1 次方程式(直接法)、固有値・固有ベクトル、フーリエ変換とその応用／時系列分析、乱数、ソート・順位付け
＜分散メモリ並列処理機能＞
　二次元複素フーリエ変換、三次元複素フーリエ変換

■ Fortran プログラムから利用する場合

Fortran プログラムから ASL ライブラリを利用する場合、リスト 18 のようにコンパイルとリンクを行います。コンパイルオプションとリンクオプションの指定方法については
表 15 をご参照ください。

リスト 18 ASL のリンク方法(Fortran)

```
シングル版・共有メモリ並列版
front1 $ sxf90 〈コンパイルオプション〉 source.f90 〈リンクオプション〉

分散メモリ並列版
front1 $ sxmpif90 〈コンパイルオプション〉 source.f90 〈リンクオプション〉
```

表 15 ASL とコンパイルオプション及びリンクオプションの指定方法（Fortran）

機能			コンパイル オプション	リンク オプション
基本機能	32 ビット整数型	シングル版		-lasl
		共有メモリ並列版	-Pmulti	-lasl64
	64 ビット整数型	シングル版		-lasl
		共有メモリ並列版	-Pmulti	-lasl64
統計機能	32 ビット整数型			-laslstat
	64 ビット整数型			-laslstat64
分散メモリ並列処理機能	dw 版		-Pmulti	-laslmpi -lasl
	ew 版		-Pmulti -ew	-laslmpiew -laslew
Fortran90 インタフェース				-laslf90 -lasl

・自動並列化を利用する場合は-Pauto、OpenMP を利用する場合は-Popenmp をコンパイルオプションにします。

リスト 19 自動並列の Fortran プログラムで 64 ビット整数型共有メモリ並列版 ASL をリンクする例

```
front1 $ sxf90 -Pauto source.f90 -lasl64
```

■ C/C++プログラムから利用する場合

C/C++プログラムから ASL ライブラリを利用する場合、以下のようにコンパイルとリンクを行います。
表 15 のオプションに加え、コンパイルオプションに「-f90lib」および「-size_t32 または-size_t64」を追加し、リンクオプションに 32 ビット整数型または dw 版を使用する場合は 「-laslcint」を、64 ビット整数型または ew 版を使用する場合は 「-laslcint64」を追加して指定してください。

リスト 20 ASL のリンク方法（C/C++）

```
シングル版・共有メモリ並列版
front1 $ sxcc 〈コンパイルオプション〉 source.c 〈リンクオプション〉
front1 $ sxc++ 〈コンパイルオプション〉 source.cpp 〈リンクオプション〉

分散メモリ並列版
front1 $ sxmpic 〈コンパイルオプション〉 source.c 〈リンクオプション〉
front1 $ sxmpic++ 〈コンパイルオプション〉 source.cpp 〈リンクオプション〉
```

リスト 21 C プログラムで 32 ビット整数型共有メモリ並列版 ASL をリンクする例

```
front1 $ sxcc -f90lib -size_t32 -Pmulti source.c -laslcint -lasl
```

■ 数学ライブラリ集 MathKeisan

MathKeisan for SX は NEC のハイパフォーマンス・コンピュータ用に高度に最適化された数学ライブラリ集です。MathKeisan に含まれるライブラリは以下のとおりです。MathKeisan のいくつかのサブルーチンについては、同機能のものが ASL にも含まれています。同機能であれば SX-ACE 用に最適化された ASL の利用をお薦めします。

■ Fortran プログラムから利用する場合

Fortran プログラムから MathKeisan ライブラリを利用する場合、リスト 22 のようにコンパイルとリンクを行います。-dw オプション(既定値)を使用してリンクする場合には、表 16 のように指定します。-ew オプションを使用してリンクする場合には、表 17 のように指定します。表に示された順番にリンクするか、またはリンク時に-hlib_cyclic オプションを指定する必要があります。

リスト 22 MathKeisan ライブラリのリンク方法（Fortran）

```
シングル版・共有メモリ並列版
front1 $ sxf90 <コンパイルオプション> source.f90 <リンクオプション>      (dw 版)
front1 $ sxf90 -ew <コンパイルオプション> source.f90 <リンクオプション> (ew 版)

分散メモリ並列版
front1 $ sxmpif90 <コンパイルオプション> source.f90 <リンクオプション>      (dw 版)
front1 $ sxmpif90 -ew <コンパイルオプション> source.f90 <リンクオプション> (ew 版)
```

表 16 MathKeisan ライブラリとリンクオプションの指定方法（-dw（規定値）を使用する場合）

MathKeisan ライブラリ	リンクオプション
BLAS	-lblas
LAPACK	-llapack -lblas
ScaLAPACK	-lscalapack -lblacsF90init -lblacs -lblacsF90init -llapack -lblas -lmpi
BLACS	-lblacsF90init -lblacs -lblacsF90init -lmpi
PARBLAS	-lparblas -Popenmp
CBLAS	-lcblas -lblas
SBLAS	-lsblas
FFT	-lfft
PARFFT	-lparfft -Popenmp
ARPACK	-larpack -llapack -lblas
PARPACK	-lparpack -larpack -llapack -lblas -lmpi

表 17 MathKeisan ライブラリとリンクオプションの指定方法（-ew を使用する場合）

MathKeisan ライブラリ	リンクオプション
BLAS	-lblas_64
LAPACK	-llapack_64 -lblas_64
ScaLAPACK	-lscalapack_64 -lblacsF90init_64 -lblacs_64 -lblacsF90init_64 -llapack_64 -lblas_64 -lmpiw
BLACS	-lblacsF90init_64 -lblacs_64 -lblacsF90init_64 -lmpiw
PARBLAS	-lparblas_64 -Popenmp
CBLAS	サポートなし
SBLAS	-lsblas_64
FFT	-lfft_64
PARFFT	-lparfft_64 -Popenmp
ARPACK	-larpack_64 -llapack_64 -lblas_64
PARPACK	-lparpack_64 -larpack_64 -llapack_64 -lblas_64 -lmpiw

■ C/C++プログラムから利用する場合

C/C++コンパイラを使って CBLAS、 BLACS 以外のライブラリをリンクする場合は、コマンドラインで MathKeisan ライブラリを指定したの後に、-f90lib オプションを付けてください。この時、-dw の MathKeisan ライブラリをリンクする場合は-f90lib dw オプションをつけ、-ew の MathKeisan ライブラリをリンクする場合は-f90lib ew オプションを付けてください。

7章 プログラムについての補足

■ 使用メモリサイズ

実行に必要なメモリサイズを事前に確認することができます。ただし allocate 等で動的に確保するメモリサイズは含まれません。

逐次プログラムの場合

リスト 23 メモリサイズの確認（逐次プログラム）

```
front1 $ sxsize 実行形式ファイル名
```

並列化プログラムの場合（自動並列または OpenMP による並列化）

リスト 24 メモリサイズの確認（並列プログラム）

```
front1 $ sxsize -fl 並列数 実行形式ファイル名
```

リスト 25 sxsize コマンド表示例

```
front1$ sxsize a.out.s （逐次プログラム）
956688(.txt) + 1274432(.data) + 542080(.bss) + 84434(.comment) + 4856(.ELNI) +
4908(.whoami) = 2867398
（逐次プログラム a.ous.s を実行すると、2,867,398 byte = 約 2.8MB 使用します）

front1$ sxsize -fl 4 a.out.p （並列プログラム）
1267584(.text) + 1287920(.data) + 457568(.bss) + 111678(.comment) + 5088(.whoami) +
1080528(logical task region) * 4 = 7451950
（並列化プログラム a.out.p を 4 並列実行すると、7,451,950 byte = 約 7.2MB 使用します）
```

■ スタックサイズの指定

並列化オプション（-Pauto/-Popenmp）オプションを付けてコンパイルしたプログラムを実行した際に、Segmentation fault で実行がエラー終了することがあります。コンパイラが推定するスタックサイズの不足が原因となることがありますので、リンク時にオプションで適切なスタック領域を確保してください。リスト 26 の例では 2GB のスタック領域を確保します。

リスト 26 スタックサイズを 2GB に指定する例

```
front1 $ sxf90 -Pauto source.f90 -Wl,-Z 2G
```

コンパイラはリンク時に以下のようにスタックサイズを見積もります。

- **並列処理用ロードモジュールの場合（-Pauto/-Popenmp）**

  スタックサイズの合計値に 35KB を加えた値

- **並列処理用でないロードモジュールの場合**

  約 96MB

■ バイナリファイルの扱い [Fortran]

データファイルを入力するなど他のサーバで作成したバイナリファイルを扱う場合、注意が必要です。SX-ACE システムは Big-Endian 仕様ですので、Little-Endian 仕様[6]のバイナリファイルを扱うには、Endian を合わせる必要があります。変換は環境変数 F_UFMTENDIAN を用い、Big-Endian に変換したい Little-Endian ファイルの装置番号を指定します。

リスト 27 環境変数 F_UFMTENDIAN

```
setenv F_UFMTENDIAN u[,u]…
```

・u は Little-Endian ファイルの装置番号です。複数ある場合は、”,”(カンマ)で区切って羅列します。

リスト 28 バッチリクエストファイル例

```
# test job.7
#PBS -q sx -b 32
setenv F_UFMTENDIAN 30,40       #装置番号 30,40 を Big-Endian に変換する
setenv MPIEXPORT “F_UFMTENDIAN”
cd $PBS_O_WORKDIR
./a.out
mpirun –np 32 –nn 32 ./a.out
```

■ バウンズチェック [Fortran]

配列の添字について、宣言している範囲外を参照していないか検査します。

プログラムの実行中に、適切な値を処理しているはずなのにエラー番号 250(オーバーフロー)や 252(ゼロ割り)、253(演算例外)等が発生する、または実行エラーは発生しないが実行する毎に動作や演算結果が異なるという場合、配列の添字検査をしてみてください。配列で参照されている添字が許される範囲にあるかどうかを調べ、範囲外であれば配列の大きさ、添字の値を表示します。

チェック方法は、-eC オプションを付けコンパイル(
リスト 29)してから実行します。範囲外添字が見つかると、標準エラー出力ファイルにエラーメッセージを出力します(リスト 30)。

リスト 29 バウンズチェックのオプション

```
front1 $ sxf90 –eC ソースファイル名
```

・-eC オプションを指定すると、最適化やベクトル化および並列化はされませんので、通常の実行に比べ実行時間が非常に長くなります。添字検査を行う時のみ、このオプションを用いてください。

リスト 30 エラーメッセージ例

```
*240 Subscript error array=b size=10 subscript=11 eln=756 PROG=main ELN=756(400021981)

配列 b はサイズ 10 の宣言であるが、添字 11 を使っているためエラーが発生している。
エラー発生箇所は、main ルーチン 756 行目。
```

[6] 本センターの並列コンピュータでは、Fortran プログラムでは環境変数で Big-Endian に設定されています。

8 章　マニュアル

■　コンパイラマニュアル

● テキスト版マニュアル

テキスト版のマニュアルはのリスト 31 のコマンドで参照できます。

リスト 31 コンパイラマニュアルの参照方法（テキスト版）

```
front1 $ sxman sxf90     (Fortran コンパイラ)
front1 $ sxman sxcc      (C コンパイラ)
front1 $ sxman sxc++     (C++コンパイラ)
```

● PDF 版マニュアル

PDF 版のマニュアルを閲覧するには並列コンピュータの SSH 接続時に X forwarding を行う必要があります。リスト 32 のコマンドを実行すると Firefox ブラウザが起動し、PDF マニュアルが閲覧できます。

リスト 32 コンパイラマニュアルの参照方法（PDF 版）

```
front1 $ mansxlangj      (日本語版)
front1 $ mansxlange      (英語版)
```

■　ライブラリマニュアル

■ ASL

リスト 33 のディレクトリに PDF 版マニュアルがあります。

リスト 33 ASL の PDF 版マニュアル

```
/usr/ap/ASL-man-super/PDF/ASL          (ASL マニュアル日本語版)
/usr/ap/ASL-man-super/PDF/ASL_e        (ASL マニュアル英語版)
/usr/ap/ASL-man-super/PDF/CINT         (ASL C 言語インタフェースマニュアル日本語版)
/usr/ap/ASL-man-super/PDF/CINT_e       (ASL C 言語インタフェースマニュアル英語版)
```

日本語マニュアルには表 18 に示したものがあります。

表 18 ASL マニュアル（日本語版）

ファイル名	タイトル	内容
1main.pdf	基本機能編 第 1 分冊	格納モードの変換,基本行列演算,最小二乗法,固有値・固有ベクトル
2main.pdf	基本機能編 第 2 分冊	連立1次方程式(直接法)
3main.pdf	基本機能編 第 3 分冊	連立1次方程式(反復法),対称連立一次方程式(反復法),非対称連立一次方程式(反復法)
4main.pdf	基本機能編 第 4 分冊	フーリエ変換とその応用／時系列分析

5main.pdf	基本機能編 第 5 分冊	微分方程式とその応用,数値微分,数値積分,3次元境界要素法用の数値積分法近似・補間,スプライン関数
6main.pdf	基本機能編 第 6 分冊	特殊関数,乱数,ソート・順位付け,方程式の根,極値問題・最適化
7main.pdf	統計機能 ASLSTAT 利用の手引	確率分布,基礎統計量,推定と検定,分散分析・実験計画,ノンパラメトリック検定,多変量解析,近似・回帰分析
8main.pdf	共有メモリ並列処理機能編	基本行列演算,連立一次方程式(直接法),固有値・固有ベクトル,連立一次方程式(反復法)フーリエ変換とその応用／時系列分析,乱数,ソート・順位付
9main.pdf	分散メモリ並列処理機能編	複素フーリエ変換

■ MathKeisan ライブラリ

並列コンピュータに SSH X Forwarding 接続し、 Firefox ブラウザを起動してリスト 34 のファイルをご参照ください。

リスト 34 MathKeisan ライブラリのマニュアル

```
/usr/ap/Mathkeisan-man/SX-ACE/J/index.html   （日本語版）
/usr/ap/Mathkeisan-man/SX-ACE/E/index.html   （英語版）
```

9 章　利用負担金

■　利用負担金について

利用負担金は、演算負担経費、ファイル負担経費、出力負担経費、可視化負担経費の 4 つがあります(表 19、表 20)。スーパーコンピュータと並列コンピュータを利用すると、演算負担経費が発生します。

共有利用は利用するノード数と経過時間によって負担額が決定し、計算資源を利用者間で共有利用する利用形態です。

また、占有利用は計算資源を待ち時間なく占有して利用することができ、申請する利用期間によって負担額が決定する利用形態です。

請求書は四半期(3 ヶ月)ごとに、利用者を取りまとめている支払責任者に発行します。最新の情報は以下の Web サイトをご覧ください。

```
http://www.ss.cc.tohoku.ac.jp/utilize/academic.html（学術利用）
http://www.ss.cc.tohoku.ac.jp/utilize/business.html（民間機関利用）
```

表 19 基本利用負担金(大学・学術利用)

区 分	項 目	利用形態	負 担 額		
演 算 負担経費	スーパー コンピュータ	共有	利用ノード数 1	(実行数、実行時間の制限有)	無料(備考2)
			利用ノード数 1～32 まで	経過時間 1 秒につき	0.06 円
			利用ノード数 33～256 まで	経過時間 1 秒につき	(利用ノード数-32)×0.002 円+0.06 円
			利用ノード数 257 以上	経過時間 1 秒につき	(利用ノード数-256)×0.0016 円+0.508 円
		占有	利用ノード数 32	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	400,000 円 720,000 円
			利用ノード数 64	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	720,000 円 1,300,000 円
			利用ノード数 128	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	1,300,000 円 2,340,000 円
	並列 コンピュータ	共有	利用ノード数 1～6 まで	経過時間 1 秒につき	0.04 円
			利用ノード数 7～12 まで	経過時間 1 秒につき	0.07 円
			利用ノード数 13～18 まで	経過時間 1 秒につき	0.1 円
			利用ノード数 19～24 まで	経過時間 1 秒につき	0.13 円
		占有	利用ノード数 1	利用期間 3 ヶ月につき (可視化システムの 20 時間無料利用を含む) 利用期間 6 ヶ月につき (可視化システムの 40 時間無料利用を含む)	160,000 円 320,000 円
ファイル 負担経費	1TB まで無料、追加容量 1TB につき年額				3,000 円
出 力 負担経費	大判プリンタによるカラープリント			フォト光沢用紙 1 枚につき クロス 1 枚につき	600 円 1,200 円
可視化 機器室利用 負担経費	1 時間の利用につき				2,500 円

表 20 基本利用負担金(民間機関利用)

区 分	項 目	利用形態	負 担 額		
演 算 負担経費	スーパー コンピュータ	共有	利用ノード数 1	(実行数、実行時間の制限有)	無料(備考2)
			利用ノード数 1～32 まで	経過時間 1 秒につき	0.18 円
			利用ノード数 33～256 まで	経過時間 1 秒につき	(利用ノード数-32)×0.006 円+0.18 円
			利用ノード数 257 以上	経過時間 1 秒につき	(利用ノード数-256)×0.0048 円+1.524 円
		占有	利用ノード数 32	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	1,200,000 円 2,160,000 円
			利用ノード数 64	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	2,160,000 円 3,900,000 円
			利用ノード数 128	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	3,900,000 円 7,020,000 円
	並列 コンピュータ	共有	利用ノード数 1～6 まで	経過時間 1 秒につき	0.12 円
			利用ノード数 7～12 まで	経過時間 1 秒につき	0.21 円
			利用ノード数 13～18 まで	経過時間 1 秒につき	0.3 円
			利用ノード数 19～24 まで	経過時間 1 秒につき	0.39 円

	並列 コンピュータ	占有	利用ノード数 1	利用期間 3 ヶ月につき （可視化システムの 20 時間無料利用を含む） 利用期間 6 ヶ月につき （可視化システムの 40 時間無料利用を含む）	480,000 円 960,000 円
ファイル 負担経費	1TB まで無料、追加容量 1TB につき年額				9,000 円
出　力 負担経費	大判プリンタによるカラープリント			フォト光沢用紙 1 枚につき クロス 1 枚につき	1,800 円 3,600 円
可視化 機器室利用 負担経費	1 時間の利用につき				7,500 円

備考

1　負担額算定の基礎となる測定数量に端数が出た場合は、切り上げる。
2　負担額が無料となるのは専用のジョブクラスで実行されたものとし、制限時間を超えた場合には強制終了する。
3　占有利用期間は年度を超えないものとし、期間中に障害、メンテナンス作業が発生した場合においても、原則利用期間の延長はしない。また、占有利用期間中のファイル負担経費は 10TB まで無料とする。
4　ファイル負担経費については申請日から当該年度末までの料金とする。

■　利用負担金の確認方法

● プロジェクトコードの確認（project コマンド）

利用負担金はプロジェクトコード毎に合算されます。利用可能なプロジェクトコードの確認、デフォルトのプロジェクトコードの設定は project コマンドをご利用ください。プロジェクトコードの名称変更、追加などについては共同利用支援係までお問い合わせください。

リスト 35 プロジェクトコードの確認

```
front1 $ project
使用可能なプロジェクトおよびキュー名は次のとおりです

利用者番号:(利用者番号)
-----------------------  使用可能なプロジェクト一覧  -----------------------

  プロジェクト名称     :  運営費交付金
  プロジェクトコード   :  un0000
  使用可能なキュー名   :  sx32 sx64 ･･･

デフォルトのジョブ実行プロジェクトは un0000 です

    1.デフォルトプロジェクト変更     9.終了

何番の処理を選びますか ？
```

● プロジェクト課金情報表示（pkakin コマンド、ukakin コマンド）

コマンドを実行した利用者が利用可能なプロジェクトについて、負担額や請求情報を表示します。負担額は前日の午前 9 時までに終了したリクエストの利用額までが反映されています。

リスト 36 プロジェクトごとの負担額、請求情報の表示

```
front1 $ pkakin
   2 月 20 日  現在の利用負担金は次のとおりです

プロジェクトコード      累計負担額      調整額累計      固定費合計      請求済額   今期請求予定額(うち請求持越額)
un0000                    15,404               0               0             0       15,404               0

支払責任者と経理担当者の所属(学校、学部)が異なる場合はセンターに連絡してください
```

```
支払費目の指定は各部局の経理担当者(学内の場合)、もしくはセンター会計係(学外の場合)へお伝えください

    1.負担金の明細表示  9.終了

何番の処理を選びますか ? 1
出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力先を選択してください  ( 1.画面  2.ファイル ) : 1

================================================================================================================
                                    プロジェクト負担金明細情報

    2 月 20 日  現在の利用額および負担額は次のとおりです

  支払責任者          :  東北 太郎
  支払責任者番号      :  aaaaaa
  プロジェクト名称    :  運営費交付金
  プロジェクトコード  :  un0000

                        合計          演算 SX          演算 LX        ファイル            出力          可視化
  利用額              15,143          14,635              508             0               0               0
  負担額              15,143          14,635              508             0               0               0
----------------------------------------------------------------------------------------------------------------
月別利用額
         2 月             15,143
----------------------------------------------------------------------------------------------------------------
利用者別利用額
        abc000               21
        abc001
           :                  6
        abc020               27
        abc021            4,895
```

リスト 37 利用者ごとの利用額情報の表示

```
front1 $ ukakin

        利用者番号=(利用者番号)

 ----------------------  利 用 額  ----------------------
  支払責任者          :  東北 太郎
  支払責任者番号      :  aaaaaa
  プロジェクト名称    :  運営費交付金
  プロジェクトコード  :  un0000

    月       演算 SX      演算 LX      ファイル         出力         可視化         合計
    4              0            0             0            0            0            0
    5              0            0             0            0            0            0
    6              0            0             0            0            0            0
    7              0            0             0            0            0            0
    8              0            0             0            0            0            0
    9              0            0             0            0            0            0
   10              0            0             0            0            0            0
   11              0            0             0            0            0            0
   12            170           10             0            0            0          180
    1            546          162             0            0            0          708
    2              0            0             0            0            0            0
    3              0            0             0            0            0            0
 合計            716          172             0            0            0          888
```

● ジャーナル(利用明細)の抜粋(ulist コマンド、plist コマンド)

コマンドを実行した利用者のジャーナル情報、プロジェクトごとのジャーナル情報を抜粋して CSV 形式(カンマ区切り)のファイルに出力します。表 21 の項目が出力されます。

リスト 38 利用者のジャーナル情報を抜粋

```
front1 $ ulist

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力するファイル名を入力してください ( 省略時:ulist.csv ) :

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

リスト 39 プロジェクトのジャーナル情報を抜粋

```
front1 $ plist

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力するファイル名を入力してください ( 省略時:plist.csv ) :
プロジェクトを選択してください

    1.(un0000)     運営費交付金

何番のプロジェクトを選びますか ? 1

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

表 21 ulist コマンドの出力項目

課金計上年月,ジャーナル連番,利用者番号,支払責任者番号,プロジェクトコード,ホスト ID,クラス ID,キュー名,投入日時,開始日時,終了日時,経過時間,使用ノード数,ユーザ CPU 時間,最大メモリサイズ,ベクトル時間,ベクトル演算率,ノード時間(使用量),利用額

● ジャーナルの集計(usum コマンド、psum コマンド)

コマンドを実行した利用者のジャーナル情報、プロジェクトごとのジャーナル情報を集計して表示します。

リスト 40 利用者のジャーナル情報を集計

```
front1 $ usum

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2

----------------------  利 用 情 報  ----------------------

利用者番号  プロジェクトコード  ホスト ID       経過時間合計  ユーザ CPU 時間合計    最大メモリサイズ  ノード時間(使用量)合計
abc001      un0000            20        15:10:00          45:55:58         61,394           18:42:08
abc001      un0000            02      1867:16:29            :  : 0          5,596             :43:03

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

リスト 41 プロジェクトのジャーナル情報を集計

```
front1 $ psum

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
```

```
終了月を入力してください : 2
プロジェクトを選択してください

    1.(un0000)     運営費交付金

何番のプロジェクトを選びますか ? 1

-----------------------  利 用 情 報  -----------------------

プロジェクトコード   ホスト ID        経過時間合計   ユーザ CPU 時間合計      最大メモリサイズ   ノード時間(使用量)合計
un0000              0         7545:43:59          :  : 0            34,816              1:25:04
un0000              20          13:13:14     2642:34:25             61,436          2171:14:38

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

■ 利用見込み額の設定

プロジェクトコードの請求先が学外の場合は、請求処理の都合上 2 月中旬以降、3 月末までの利用額を見込み金額として設定します。設定した金額は、1 月から 2 月中旬までの利用負担金に合算し、2 月末に請求いたします。見込み金額の設定は mikomi コマンドで行います。

リスト 42 mikomi コマンドの例

```
front1 $ mikomi

プロジェクトを選択してください
プロジェクトコード : プロジェクト名称  支払責任者番号 : 支払責任者氏名
un0000        : 運営費交付金        aaaaaa      : 東北 太郎

プロジェクトコードを入力してください un0000

 2 月 1 日 現在の見込み額は次のとおりです

  支払責任者      : 東北 太郎
  支払責任者番号 : aaaaaa
  プロジェクト名称 : 運営費交付金
  プロジェクトコード: un0000
  見込み額指定者 :

  見込み額                :         0 円
  今期請求予定額          :   300,000 円
  合計請求額(見込み額込) :   300,000 円

 1.見込み額の指定 2.削除 9.終了
 何番の処理を選びますか ? 1

  見込み額を入力してください (円) ? 200000
  登録してよろしいですか (yes/no) ? yes

 2 月 1 日 現在の見込み額は次のとおりです

  支払責任者      : 東北 太郎
  支払責任者番号 : aaaaaa
  プロジェクト名称 : 運営費交付金
  プロジェクトコード: un0000
  見込み額指定者 : 東北 太郎 (利用者番号 変更日: 2 月 1 日)

  見込み額                :     200,000 円
  今期請求予定額          :     500,000 円
  合計請求額(見込み額込) :     500,000 円

 見込み額の指定 2.削除 9.終了
 何番の処理を選びますか ?
```

10 章　運用に関する情報、お問い合わせ

■　運用に関するお知らせ

大規模科学計算システムのホームページで利用方法やシステムの運用状況について、最新情報をお知らせします。計画停電による運用の停止スケジュールなどについても、こちらでお知らせしております。

URL http://www.ss.cc.tohoku.ac.jp/

■　利用申請

利用申請に関する詳細は、以下のウェブページをご覧ください。申請書の提出、お問合せは共同利用支援係までお願いいたします。

URL http://www.ss.cc.tohoku.ac.jp/utilize/index.html
メールアドレス　uketuke@cc.tohoku.ac.jp　**電話番号**　022-795-3406

■　利用に関するお問い合わせ

■ システムの利用方法についてのお問合せ

利用相談員が対応いたします。以下のメールアドレスまでお問合せください。

メールアドレス　sodan05@cc.tohoku.ac.jp

■ 利用負担金についてのお問合せ

共同利用支援係までお問い合わせください。

メールアドレス　uketuke@cc.tohoku.ac.jp　**電話番号**　022-795-3406

■ 請求書についてのお問合せ

請求書の発送等については会計係までお問い合わせください。

メールアドレス　kaikei@cc.tohoku.ac.jp　**電話番号**　022-795-3405

■ その他のお問合せ

総務係までお問い合わせください。

メールアドレス　som@cc.tohoku.ac.jp　**電話番号**　022-795-3407

おわりに

本稿では、SX-ACE システムのプログラミング利用ガイドとして基本的な手順を紹介しました。研究室のサーバでは実現できなかったプログラムやアイデアを、ぜひ最新鋭のスーパーコンピュータシステムでお試しください。研究の強力なツールとしてご活用いただければ幸いです。ご不明な点、ご質問等ございましたら、お気軽にセンターまでお問い合わせください。